THE COMPATIBILITY OF PATENT LAW AND THE INTERNET

*Jeanne C. Fromer**

Much ink has been spilled and many bits have been used discussing what the Internet's architecture and values ought to mean for the future of copyright law.[1] And though much has been written about the patentability of software,[2] how, if at all, patent law and the Internet's values are compatible is undertheorized.[3] Through the lens of recent books by Jonathan Zittrain, *The Future of the Internet—And How To Stop It*,[4] and David Post, *In Search of Jefferson's Moose: Notes on the State of Cyberspace*,[5] I explore this issue. Although a central value of the Internet is inclusiveness and the patent right is directed to exclusivity, I suggest that if tailored appropriately, patent law can be supportive of the Internet's core values.

Part I discusses theories of the Internet, as put forth in the books by Zittrain and Post. Part II turns to the patent system. Part III weaves together the two topics to explore the compatibility of patent law and the Internet.

I. THE INTERNET

Jonathan Zittrain tells an engaging story in his book of how computers and the Internet came to be and what they might one day become. He suggests that the source of that which we value in software and the Internet—its generativity—can also be cause for the Internet's failure.[6] He coins and defines "generativity" to mean "a system's capacity to produce

* Associate Professor, Fordham Law School. I thank Arnaud Ajdler, Sonia Katyal, and the participants in this Symposium for their helpful comments.

1. *E.g.*, YOCHAI BENKLER, THE WEALTH OF NETWORKS: HOW SOCIAL PRODUCTION TRANSFORMS MARKETS AND FREEDOM (2006); LAWRENCE LESSIG, FREE CULTURE: THE NATURE AND FUTURE OF CREATIVITY (2005); JESSICA LITMAN, DIGITAL COPYRIGHT (2006); Jane C. Ginsburg, *"The Exclusive Right to Their Writings": Copyright and Control in the Digital Age*, 54 ME. L. REV. 195 (2002); Mark A. Lemley & R. Anthony Reese, *Reducing Digital Copyright Infringement Without Restricting Innovation*, 56 STAN. L. REV. 1345 (2004).

2. *Infra* sources cited in Part III.

3. One notable exception is BENKLER, *supra* note 1, which tends to be critical of strong patent laws in the face of information production.

4. JONATHAN ZITTRAIN, THE FUTURE OF THE INTERNET—AND HOW TO STOP IT (2008).

5. DAVID G. POST, IN SEARCH OF JEFFERSON'S MOOSE: NOTES ON THE STATE OF CYBERSPACE (2009).

6. ZITTRAIN, *supra* note 4, at 1–5.

unanticipated change through unfiltered contributions from broad and varied audiences."[7] Zittrain associates five factors with generativity:

> (1) how extensively a system or technology leverages a set of possible tasks; (2) how well it can be adapted to a range of tasks; (3) how easily new contributors can master it; (4) how accessible it is to those ready and able to build on it; and (5) how transferable any changes are to others.[8]

The same generativity, then, that yields for so many the Internet's pleasures of, say, Google Earth, digital music and television episodes on demand, and Wikipedia, also enables hackers to steal these same people's credit cards or other personal details, crash their software, and bring down networks.

After colorfully laying out the history of computing and the Internet and how generativity is central to the success of each,[9] Zittrain dedicates the last third of his book to exploring the "How To Stop It" segment of the book's title. What seems to animate Zittrain's proposed solutions is his desire to preserve and foster good generativity, while quashing the bad kind. He is worried about a future in which the good generativity is stamped out along with the bad, with companies providing tethered—but safe—information appliances designed with firmware or software to perform only particular specified functions (such as a digital music player, a GPS system, and even a digital toaster). Tethered appliances, according to Zittrain, would perform their specified function well but would not be generative principally because they would not be adaptable to other tasks and would not easily be built upon.[10] That is, the music player would not make toast, nor would the toaster be able to give GPS-based directions or be built to toast bread in some new way. General-purpose computers could in theory do all three tasks and then some. In light of general-purpose computers' generativity, Zittrain seeks to offer another way to the future. For example, he suggests that general-purpose computers might have a safe "green" zone to store important data and trustworthy software and a more risky "red" zone on which to experiment with other software.[11] According to his proposal, some "red" zone software might turn out to be harmful but removable before damaging anything in the "green" zone, while some might become reliable and beneficial enough for subsequent inclusion in the "green" zone.[12] Most centrally, Zittrain suggests that Internet users need to be provided with more and better information about which Internet applications and sites are reliable to use without infecting the user's computer with a virus or hacking the user's data.[13]

---

7. *Id.* at 70 (emphasis omitted).
8. *Id.* at 71.
9. *Id.* at 7–148.
10. *Id.* at 101–03.
11. *Id.* at 154–57.
12. *Id.*
13. *Id.* at 157–62.

David Post takes a different tack than Zittrain. He writes about the Internet's development in the context of Thomas Jefferson's experiences and thoughts, which at first glance seems near-Mesozoic compared with the contemporary fast-paced growth of the Internet. Post nonetheless establishes the analogy's fit, suggesting that Jefferson's animating republican beliefs—a penchant for self-governing, but interlinked, communities, rather than strong, centralized government—is precisely what has made the Internet such a success thus far and ought to be preserved. As one of many examples of Jefferson's republicanism, Post points to Jefferson's approach to the over 800,000 square miles of land acquired in the Louisiana Purchase: let subterritories within the purchase govern themselves, with the power to petition to become states of the United States down the line.[14] As Jefferson explained, "I have much confidence that we shall proceed successfully for ages to come, and that . . . it will be seen that the larger the extent of country, the more firm its republican structure, if founded, not on conquest, but in principles of compact and equality."[15]

Post applies these Jeffersonian principles to the story of the Internet's development, indicating how time and time again, the Internet was structured in a decentralized fashion, which allowed for its exponential growth. For example, Post describes the Internet protocol for transferring data over the Internet, TCP/IP.[16] In simplified form, TCP/IP works to transport raw data from users' applications (such as e-mail programs or Web browsers) to their destination. The protocol does not concern itself with what the data represent—be they part of a text message, a picture, sound, or something else—but just carries the bits associated with a particular application. In that sense, TCP/IP is decentralized because it allows for any sort of application—even those developed after the protocol's development—to link up to it for sending data because the protocol does not need to know what the data mean.[17]

This protocol is essential to the Internet's notion of end-to-end computing, by which, according to Post, the processing of data through varied applications happens at the endpoints of the Internet—the users' computers—while the shuffling around of uncomprehended bits happens over the network. As Post puts it,

> Smart machines, connected to a dumb network. Complicated and sophisticated applications, and a network doing nothing more than moving bits around as directed by those applications. That's the Internet. All the interesting stuff is at the edges—the network just gets the bits there, as quickly and efficiently as possible.[18]

14. POST, *supra* note 5, at 115–16.
15. *Id.* (emphasis omitted) (quoting Jefferson).
16. *Id.* at 25. TCP/IP stands for Transmission Communication Protocol/Internet Protocol. *Id.*
17. *Id.* at 80–83.
18. *Id.* at 86.

To Post, the Internet—as grounded in end-to-end computing—is very much like Jefferson's notion of a loosely linked community, with self-governing subgroups.

Both Zittrain's and Post's books hold out the wondrous promise of the Internet as a tool of communication, knowledge, computation, scientific research, self-governance transcending our physical governmental boundaries, and future applications we cannot begin to imagine today. Each book focuses in great detail on how the Internet came to be and its vision of the guiding principles for the Internet going forward. Zittrain emphasizes generativity as a key aspect of the Internet worth preserving, while Post highlights the Internet's decentralized and self-governing aspects.

Each book dedicates a small amount of attention to an essential legal issue affecting the Internet's future, namely, the patent protections available to cover varying aspects of that technology. Depending on how the patent laws are structured, they might grant incentives to innovate with regard to the Internet (if the protections are structured just right), stifle innovation in the area (if the protections are granted too easily or not easily enough), or some mixture of both (if the protections accord just the right incentive to some subsets of innovation in the area, but not to others).

Zittrain sets out two competing types of software production: proprietary software, which is often protected by patents, and open-source software, which is not.[19] He relays the concern that, for the former class of software production, perhaps too many software patents issue, and to large firms at that, for the purpose of "extract[ing] royalties from rivals and to defend themselves from their rivals' patents," thereby causing the unfortunate situation of a patent thicket.[20] He expresses worry at this situation for numerous reasons: smaller firms and individuals may not be included in cross-licensing if they do not also have software patents, and infringement can occur even if the software developer did not know of the patented invention and even if the patented invention's specific code is not used, so long as the same abstract concepts are present in each.[21] Zittrain suggests that this situation might not be so bad so long as many actual instances of patent infringement—which he thinks will occur quite often—are tolerated.[22] In addition, he proposes that more infringing uses (or close questions of such) be tolerated by making it harder for patentees to sue for

19. ZITTRAIN, *supra* note 4, at 189.

20. *Id.* at 190 (citing James Bessen & Robert M. Hunt, *The Software Patent Experiment*, 14–15 (Research on Innovation Working Paper, 2004), *available at* http://www.researchoninnovation.org/softpat.pdf).

21. *Id.*

22. *Id.* at 190–91.

infringement, in part by modifying the statute of limitations and by reinvigorating the doctrine of laches.[23]

Post has a different take on the role of patent law in the Internet's development. He focuses on Jefferson's central role in American patent law, as author of the first national patent law, first Commissioner of Patents, and inventor of such varied items as a weather vane, folding chair, automated calendar, and plow.[24] Post relays Jefferson's skepticism about strong patent protection:

> That ideas should freely spread from one to another over the globe seems to have been peculiarly and benevolently designed by nature when she made them, like the air we breathe, incapable of confinement or exclusive appropriation, and, like fire, expansible over all space without lessening their density in any point.[25]

Post recognizes that intellectual property protection is about trade-offs between the benefits the protection can bring, in the form of increased innovation, and the costs the protection imposes on society, in terms of higher costs and decreased competition.[26] Like Jefferson, Post is skeptical about many patent grants, in that people often feel the need to create valuable inventions regardless of the availability of patent protection.[27] Post indicates, without much more detail, that patent law ought to be about drawing the line so that the benefits of patent protection outweigh the costs.[28] He speculates that so much Internet-related innovation, such as open-source software, happened without intellectual property protection,[29] intimating that the law ought to be cautious and not make patent protection for Internet-related innovation too strong.

Zittrain and Post, then, both appear to be skeptical of applying patent law to Internet-related innovation, even while they suggest that it might serve some beneficial role. This essay picks up their ruminations on patent protection for Internet-based inventions and explores whether patent law is compatible with the Internet's central goals, as suggested by Zittrain and Post. I conclude that they can be compatible if care is taken to tailor the application of patent law to encourage the sort of software innovation underlying much of the Internet's success. Before turning to that, I provide some background on patent law in the next part.

---

23. *Id.* at 191–92.

24. POST, *supra* note 5, at 194–96. Other scholars suggest that Thomas Jefferson ought not to be viewed as the central historical figure in American patent law. *See, e.g.*, Adam Mossoff, *Who Cares What Thomas Jefferson Thought About Patents?: Reevaluating the Patent "Privilege" in Historical Context*, 92 CORNELL L. REV. 953 (2007).

25. POST, *supra* note 5, at 197 (emphasis omitted) (quoting Jefferson).

26. *Id.* at 198–200. Decreased competition may result in both the protected invention and any follow-up innovation that does not occur because of the protection. *Id.*

27. *Id.* at 200.

28. *Id.* at 200–01.

29. *Id.* at 203.

II. PATENT LAW

The principal goal of the U.S. patent system is to stimulate innovation,[30] as manifested in the Constitution's articulation of Congress's power "[t]o promote the Progress of . . . useful Arts, by securing for limited Times to . . . Inventors the exclusive Right to their . . . Writings and Discoveries."[31] In theory, this stimulation occurs by rewarding inventors with a time-limited, exclusive patent right for taking two steps they likely would not otherwise take: to invent in the first instance[32] and to reveal information to the public about these inventions,[33] thereby enriching society with the invention and the ability to build on the invention.[34]

An inventor can obtain a patent so long as he demonstrates that his invention or discovery is patentable subject matter—a "new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof"[35]—and is novel,[36] useful,[37] and nonobvious.[38] Patents are granted after successfully undergoing examination by the Patent and Trademark Office to ascertain that an invention meets patentability conditions and its description in the patent application satisfies specified disclosure requirements[39] of a written description, enablement, and best mode.[40] The written description requirement ensures that the inventor is in possession of the claimed invention.[41] To enable the invention, the patent applicant must demonstrate in the specification to "any person skilled in the [relevant] art [how] . . . to make and use the [invention],"[42] without "'undue experimentation.'"[43] Also, the patent applicant must set out "the best mode contemplated by the inventor of carrying out his invention."[44] The best-mode requirement is met so long as the patent document objectively discloses the best mode that the inventor subjectively conceived by the time

30. Dan L. Burk & Mark A. Lemley, *Policy Levers in Patent Law*, 89 VA. L. REV. 1575, 1597–99 (2003).

31. U.S. CONST. art. I, § 8, cl. 8.

32. Burk & Lemley, *supra* note 30, at 1581–82 (analyzing the costs of research and development across various industries).

33. Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 480–81 (1974) (discussing patent law's disclosure requirements and the policy concerns motivating such requirements).

34. Jeanne C. Fromer, *Patent Disclosure*, 94 IOWA L. REV. 539 (2009).

35. 35 U.S.C. § 101 (2006).

36. *Id.* § 102.

37. *Id.* § 101.

38. *Id.* § 103.

39. *Id.* § 131.

40. *Id.* § 112.

41. Guang Ming Whitley, Comment, *A Patent Doctrine Without Bounds: The "Extended" Written Description Requirement*, 71 U. CHI. L. REV. 617, 628–29 (2004).

42. 35 U.S.C. § 112.

43. Monsanto Co. v. Syngenta Seeds, Inc., 503 F.3d 1352, 1360 (Fed. Cir. 2007) (quoting *In re* Wright, 999 F.2d 1557, 1561 (Fed. Cir. 1993)) (citing *In re* Vaeck, 947 F.2d 488, 495 (Fed. Cir. 1991); *In re* Wands, 858 F.2d 731,736–37 (Fed. Cir. 1988); *In re* Fisher, 427 F.2d 833, 839 (C.C.P.A. 1970)).

44. 35 U.S.C. § 112.

the inventor files the patent application.[45] Once granted, the patent right permits the patentee to exclude others from practicing the invention claimed in his patent for a term of approximately twenty years.[46]

Neither Congress nor the U.S. Supreme Court has issued the definitive word on the preliminary question of whether software inventions, such as those at the heart of the Internet's success, constitute patentable subject matter. In 1972, in *Gottschalk v. Benson*,[47] the Supreme Court ruled that a method for converting binary-coded decimal numbers into pure binary numbers is unpatentable.[48] The Court invoked its long-standing rule that "[p]henomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable, as they are the basic tools of scientific and technological work."[49] The rule's asserted justification is that certain basic building blocks of science and technology ought not to be patentable because they are central to further innovation and ought to be freely available. Applying this rule, the Court concluded that the method was not patentable because it was an abstract concept as well as a basic building block and because a patent on the method would wholly preempt any use of the concept, useful only in computers.[50]

Benson's rule led to a next wave of software patents—for inventions of classic patentable classes of matter—most pertinently industrial processes in which software played a role and machines containing software as an integral part. In 1982, in *Diamond v. Diehr*,[51] the Supreme Court approved of the patentability of a process of the former type to calculate the optimal time for curing rubber, using in part a computer employing Arrhenius's equation.[52] The Court noted that the patentee was not seeking to preempt all uses of the equation, but rather to embed the algorithm as a component in an industrial process just to cure synthetic rubber.[53] In 1996, the U.S. Court of Appeals for the Federal Circuit—the appellate court with exclusive jurisdiction over almost all patent appeals[54]—held that software inventions claimed as parts of a general-purpose computer constituted patentable subject matter as well, because they produce "'a useful, concrete and tangible result.'"[55] The Federal Circuit revisited this decision in 2009 in *In*

45. Eli Lilly & Co. v. Barr Labs., Inc., 251 F.3d 955, 963 (Fed. Cir. 2001).
46. 35 U.S.C. § 154(a).
47. 409 U.S. 63 (1972).
48. *Id.* at 71–72.
49. *Id.* at 67.
50. *Id.* at 68–72.
51. 450 U.S. 175 (1981).
52. *Id.* at 187–88.
53. *Id.*
54. 28 U.S.C. § 1295(a)(1) (2006); Holmes Group, Inc. v. Vornado Air Circulation Sys., Inc., 535 U.S. 826, 829–30 (2002) (carving out a limited exception for cases in which the patent issue arises only in a counterclaim).
55. State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1373 (Fed. Cir. 1998) (quoting *In re* Alappat, 33 F.3d 1526, 1544 (Fed. Cir. 1994)).

re Bilski,[56] stating that the proper test for deciding whether a process is patentable is based on the Supreme Court's prior case law in *Benson* and *Diehr*, which asked whether the process is tied to a machine or "transforms [an] article into a different state."[57] This test leaves open whether software claimed as part of a general-purpose computer is patentable because the court did not answer whether a general-purpose computer programmed to run particular software is a new machine.[58] Though the Supreme Court has granted certiorari in *Bilski*,[59] it can decide the case without speaking to software's patentability, as the invention at issue does not involve software. The patentability of software running on a general-purpose computer is thus currently up in the judicial air.

The question of whether such software inventions—often, the sort of innovation related to the Internet—constitute patentable subject matter is addressed to how worthwhile it is to test software inventions on an individualized basis for novelty, nonobviousness, utility, and adequate disclosure, rather than to make a class-based decision that almost all such inventions would not be patentable. In the next part, I explore whether software invention as a whole ought to be outside patentability and, if not, whether software inventions can fulfill the other patentability criteria. I wade into that discussion through the context of Zittrain's and Post's books.

III. THE COMPATIBILITY OF PATENT LAW AND THE INTERNET

In his book, Zittrain highlights generativity as being at the core of the Internet's success and worries in some part that patent law's valuable incentives to innovate may be overshadowed by extensive infringement liabilities, impliedly impeding the Internet's generativity. Post emphasizes decentralization and end-to-end computing as key to the Internet's success and worries that patent law might exact too many costs to be worth its general availability for software inventions. In so doing, each author picks up on some significant scholarly strands on software patentability. I argue herein that general patentability for software inventions, however, can promote generativity and end-to-end computing, so long as the law accounts for worries akin to those of Zittrain and Post.

Zittrain properly recognizes that the promises and dangers of the Internet lie in its generativity. Finding solutions to prevent the dangers without stamping out generativity is something at which he takes some honorable stabs. One such example is the StopBadware project, started at the Harvard University Berkman Center, which provides information to help users avoid

56. 545 F.3d 943 (Fed. Cir. 2008) (en banc), *cert. granted sub nom.* Bilski v. Doll, 129 S. Ct. 2735 (2009) (No. 08-964).

57. *Id.* at 958–63.

58. *Cf. id.* at 976–98 (Newman, J., dissenting) (arguing that software-implemented processes are patentable).

59. 129 S. Ct. 2735.

harmful software.[60] Zittrain's proffered solutions, however, seem like preliminary thoughts requiring more robust development. This critique is in effect not of Zittrain, but is rather a celebration of the generativity and decentralized developments critical to the Internet's successes. That is, perhaps the seeds of the robust solutions to the Internet's worries are being sowed right now in varied—perhaps complementary—ways in Beijing, Boise, and Bilbao. Certain solutions could come from a computer science graduate student experiencing a "Eureka!" moment or from a researcher working on a project at a large, established software company. What shape these solutions would take cannot now be known, but it is nearly a certainty that many attempts will come, just as Google's search engine, Wikipedia, online shopping, and many other forms of Internet software were developed in varied ways to create what we now consider to be the Internet. This much can be appreciated by Post's tale of end-to-end computing, with intelligence at the level of individual machines, which can then be distributed all over the Internet.

Certainly only a minimal fraction of contributions will be made by well-funded and nonprofit organizations like the Berkman Center and the StopBadware project, which underscores how important the availability of intellectual-property protection is. Evidence shows—as Post suggests—that creatively inclined individuals feel compelled to create, regardless of whether there is intellectual property protection available at the end of the creation rainbow.[61] Moreover, psychologists emphasize that "[p]eople will be most creative when they feel motivated primarily by the interest, enjoyment, satisfaction, and challenge of the work itself—not by external pressures."[62] Nonetheless, even an individual who feels compelled to create would nearly always have insufficient time and attention to do so if the person's work could be freely copied. Without the chance of intellectual property protection, the creator cannot typically earn a living being creative. Moreover, the individual's work does not publicize and distribute itself, even in the age of the Internet. It is therefore essential that organizations and individuals provide creators with support for professional success by paying for, promoting, marketing, and distributing their works. These vital institutional mechanisms will arise around creative people only if there is a proper economic motivation for them to do so, as with the possibility of intellectual property protection to reward successful creativity by preventing copying of the protected works.[63]

---

60. StopBadware, http://www.stopbadware.org (last visited Apr. 18, 2010).

61. *E.g.*, MIHALY CSIKSZENTMIHALYI, CREATIVITY: FLOW AND THE PSYCHOLOGY OF DISCOVERY AND INVENTION 107–26 (1996) (discussing how creative people love what they do and create for the state of flow it produces, rather than for money).

62. Beth A. Hennessey & Teresa M. Amabile, *The Conditions of Creativity*, *in* THE NATURE OF CREATIVITY: CONTEMPORARY PSYCHOLOGICAL PERSPECTIVES 11, 11 (Robert J. Sternberg ed., 1988).

63. *Cf.* Diane Leenheer Zimmerman, *It's an Original! (?): In Pursuit of Copyright's Elusive Essence*, 28 COLUM. J.L. & ARTS 187, 189–90 (2005) ("For [some], the function of

Moreover, numerous conditions in the current software industry suggest that patent protection would frequently be welcome and not harmful. Despite Post's emphasis on the fact that many key aspects of the Internet's development happened without patent seeking, the software industry has changed since its early days. There are increasing barriers to entry, like network effects diminishing the ease of innovation in the industry.[64] Moreover, Ronald Mann indicates that "direct evidence of high R&D spending in the software industry undermines claims that software patents cause firms to reduce R&D spending."[65] Finally, based on extensive interviews with those in the industry, Mann also shows that the existence of extensive software patent portfolios in the hands of large, incumbent firms does not significantly constrain the development of newer software firms, as software patents can be difficult to appropriate and infringement hard to detect.[66]

The foregoing analysis suggests that preserving the potential for generativity requires creating a legal environment that maximizes the varieties of creativity in software contributions and contributors. I argue that a carefully calibrated patent law is a good fit for encouraging a considerable subset of these possible contributions.

As an initial matter, software inventions as a class ought to be considered patentable subject matter. Software inventions fit in the mainstream of the classes of inventions protected by patent law.[67] As Pamela Samuelson, Randall Davis, Mitchell Kapor, and Jerry Reichman demonstrate, software inventions are—like patentable subject matter more generally—principally directed to their functionality.[68]

---

the law is to structure a sector of the economy so that those who participate in the chain that stretches from the origination to the dissemination of expressive works—be they creative or purely informational, high authorship or low—can reap the rewards of their toil.").

64. Pamela Samuelson, Randall Davis, Mitchell D. Kapor & J.H. Reichman, *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308, 2312 (1994).

65. Ronald J. Mann, *Do Patents Facilitate Financing in the Software Industry?*, 83 TEX. L. REV. 961, 962 (2005) (disputing evidence offered by James Bessen and Robert M. Hunt in a manuscript later published as *An Empirical Look at Software Patents*, 16 J. ECON. & MGMT. STRATEGY 157, 173–74, 180–84 (2007)).

66. *See id.* at 962, 978–79, 999–1009.

67. *See* Jeanne C. Fromer, *A Psychology of Intellectual Property*, 104 NW. U. L. REV. (forthcoming 2010) (suggesting that software inventions fit more naturally in patent law, which focuses on rewarding problem solving, than in copyright law, which focuses on rewarding problem finding).

68. Samuelson, Davis, Kapor & Reichman, *supra* note 64, at 2312 (arguing, nonetheless, against the patentability of software and in favor of *sui generis* protection for software to guard against the fact that "behavior and other industrial design elements of [software] programs are often expensive to develop and inexpensive to copy"). Samuelson et al. rightly reject the centrality of copyright law in protecting software innovation, as copyright law is directed to protection of expression, which software code is, but "the most important property of programs is their behavior (i.e., the set of results brought about when program instructions are executed)." *Id.* at 2314.

John Allison, Abe Dunn, and Ronald Mann show further that a significant number of types of software creators—such as firms making software products as opposed to services and many start-up companies—have good cause to seek patents as a return for their functional creations. After working through empirical data of patenting by firms in *Software Magazine*'s "Software 500" from 1998–2002, they suggest that "the ability to obtain patents on software always has been important to some of the industry incumbents, while others have exhibited little need for patents and displayed, in some cases, strenuous opposition to the patentability of software."[69] Their data reveal that firms offering off-the-shelf, noncustomized software programs rely more heavily on patent protection than do firms offering customized software services.[70] Patent protection can be crucial for the former type of firm, as software is "often expensive to develop and inexpensive to copy."[71] They explain how patents are less important for the latter:

> Patents seem likely to be a relatively more effective tool for protecting innovation in products than in services. To the extent a firm can provide a unique level of skilled services, it may be feasible to maintain much of the differentiating knowledge in a tacit form, bound up with the skills of the individual employees. Conversely, a products firm that sends its product out into the marketplace in many instances will be vulnerable to appropriation by competitors.[72]

Patents can also be valuable for startup software companies to prevent larger firms from copying their software and to signal to potential investors that they are financially and technologically worthy of venture funding.[73] Relatedly, many independent inventors are poorly placed to package their software inventions and to transact with larger incumbent firms over their software contributions, which is in part why a number of technology

---

69. John R. Allison, Abe Dunn & Ronald J. Mann, *Software Patents, Incumbents, and Entry*, 85 TEX. L. REV. 1579, 1580 (2007).

70. *Id.* at 1600–02.

71. Samuelson, Davis, Kapor & Reichman, *supra* note 64, at 2312.

72. Allison, Dunn & Mann, *supra* note 69, at 1601 (citing Mann, *supra* note 65, at 985; Samuelson, Davis, Kapor & Reichman, *supra* note 64, at 2333–39). Allison, Dunn, and Mann recognize the "ambiguity of causation" in the finding, in that it is unclear whether product firms patent more frequently because they make software products or whether the fact that they sought patents allowed them to survive as software product firms. *Id.* at 1603. They also show that those areas with the fewest software firms—that is, those lacking industry concentration—correlate significantly with propensity to patent, perhaps because it is plausible for them to stake out their territory and repel would-be competitors. *Id.* at 1606 & fig.8.

73. *Id.* at 1609–11; Robert P. Merges, *Software and Patent Scope: A Report from the Middle Innings*, 85 TEX. L. REV. 1627, 1657 (2007). These patents are, perhaps surprisingly, not typically used in litigation against larger companies so as not to poison potential future business associations, but instead used as bargaining chips in cross-licensing one another's patents. Allison, Dunn & Mann, *supra* note 69, at 1610–11.

licensing companies, often "labeled pejoratively as 'trolls,'" have arisen to act as middlemen between these independent inventors and large firms.[74]

Given how significant the presence of these patent-seeking classes of software producers are in the industry, there is a strong argument for making available protection for at least some aspects of functionality, as patent law does. Otherwise, these contributions might never have been made.

Contrary to Zittrain's worries, patent availability gives small firms and individuals a way to play in the big leagues by enabling them to secure patents, which they could use to cross-license and gain access to all sorts of technology developed by other entities.[75] That is, patents can give smaller and newer software players the leverage they need to secure entrée to important mainstream technologies on which to build and connect.[76] Patent law, if effectuated properly, could thus help democratize the playing field in the software industry by minimizing barriers to entry.

Moreover, the tethered appliances Zittrain so fears are more likely to proliferate in the absence of patent protection. In a world without patents, firms that would otherwise prefer patenting would tend to favor tethered appliances over general software applications, as they—and their oft-specialized hardware—would be more costly and difficult to copy or appropriate than software executed on general-purpose computers, with its readily available decompilers to reverse engineer software.

Nonetheless, patent protection is not necessary for all Internet software producers. Many companies providing open-source software products, a structure that has been politically antithetical to software patenting, have the capability to offer services structured in such a way that patent protection is principally inessential.[77] Allison, Dunn, and Mann explain why. They indicate that

> the commercially successful open-source programs share the salient characteristic that they benefit from extensive financial support from large incumbent firms [like IBM]. The firms making those investments have done so as part of a "value-chain" strategy, in which the firms seek to commoditize a part of a value chain in which they are unlikely to dominate (like the operating system), hoping to extract value at some other part of a value chain (like the servers on which the operating system

74. Allison, Dunn & Mann, *supra* note 69, at 1612–16 (citing John M. Golden, Commentary, *"Patent Trolls" and Patent Remedies*, 85 TEX. L. REV. 2111, 2135 (2007); Joe Beyers, *Rise of the Patent Trolls*, CNET NEWS, Oct. 12, 2005, http://news.com.com/rise+of+the+patent+trolls/2010-1071_3-5892996.html). Perhaps the biggest concern with these companies is that they are not worried about countersuits in infringement litigation, as they make no products, and thus the risks of bringing suit for infringement are lower—perhaps too low—than for other software patentees. Mann, *supra* note 65, at 1023.

75. Merges, *supra* note 73, at 1657.

76. Fromer, *supra* note 34, at 556–57.

77. Allison, Dunn & Mann, *supra* note 69, at 1611.

> runs, the middleware that runs on the stack above the operating system, or the services necessary to assemble all of those pieces into a well-designed "solution").[78]

Moreover, many of the successful open-source firms offer customizable software services and thus have less need for patents, as they can retain tacit know-how as a form of protection.[79]

Throwing open to patenting the software that is instrumental to the generativity and end-to-end computing of the Internet raises significant concerns that a robust patent law must address. Most notably, patent protection might seem to run counter to the Internet's principal goals of inclusivity by offering up a right to exclude others from patented software inventions. This exclusion might impede the incremental advancements in applications and services characterizing the software industry. In fact, this worry brings Samuelson, Davis, Kapor, Reichman, and others to argue that software ought not to be patented.[80]

There are a number of ways to tailor patent law to ameliorate this and other concerns.[81] For one thing, patent law ought to require more sufficient disclosure of patented software inventions. There is systemic inadequate disclosure in software patents,[82] given that the Federal Circuit seems to think erroneously that minimal disclosure of software's functionality is sufficient to enable those skilled in the art to encode the invention.[83] Disclosure of a software invention's functionality is not enough to ease incremental development of software. Disclosing actual source code—or at least detailed flow charts and the like—would more readily contribute to the database of knowledge of software solutions to various problems.[84] While patent protection would prevent the free use of all such disclosed code,

---

78. *Id.* (citing Ronald J. Mann, *Commercializing Open Source Software: Do Property Rights Still Matter*?, 20 HARV. J.L. & TECH. 1, 12, 24–25 (2006)).

79. *Id.* That said, there might be some political shifts in this regard, as some open-source companies, such as Linux distributor Red Hat, have started to file for patent protection. Red Hat Patent Policy, http://www.redhat.com/legal/patent_policy.html (last visited Apr. 18, 2010).

80. Samuelson, Davis, Kapor & Reichman, *supra* note 64, at 2345–46; Richard M. Stallman, The Dangers of Software Patents, Address at the University of Dublin, Trinity College (May 24, 2004), http://www.ifso.ie/documents/rms-2004-05-24.html.

81. *Cf.* Julie E. Cohen & Mark A. Lemley, *Patent Scope and Innovation in the Software Industry*, 89 CAL. L. REV. 1, 3 (2001) (arguing that patent law ought to be available to software inventions, though the law ought to be tailored to these inventions' particular characteristics).

82. Fromer, *supra* note 34, at 584; Jeanne C. Fromer, *The Layers of Obviousness in Patent Law*, 22 HARV. J.L. & TECH. 75, 96–97 (2008).

83. *E.g.*, N. Telecom, Inc. v. Datapoint Corp., 908 F.2d 931, 942 (Fed. Cir. 1990) ("'[T]he conversion of a complete thought (as expressed in English and mathematics, i.e. the known input, the desired output, the mathematical expressions needed and the methods of using those expressions) into the language a machine understands is necessarily a mere clerical function to a skilled programmer.'" (alteration in original) (quoting *In re* Sherwood, 613 F.2d 809, 817 n.6 (C.C.P.A. 1980))).

84. Fromer, *supra* note 34, at 576 n.160.

interested parties would have free access in the patent itself to review the code. They could then freely appropriate the code's unprotected components, license code reflecting protected elements, or use the code as a basis to conceptualize other ways of creating particular software functionality, thereby avoiding wasteful efforts to reinvent the digital wheel time and time again.

Moreover, antitrust law and related principles in patent law would have an important role to play in ensuring that patent law would not operate to keep would-be competitors out of areas in which software-based standards have emerged or are emerging on which patents are held.[85] Relatedly, Julie Cohen and Mark Lemley suggest a limited right to reverse engineer patented software inventions in order to study and copy their unprotected elements, which would help ensure interoperability in a world where network effects and standards take hold.[86] Had software patents been more clearly allowed at the Internet's infancy, such legal principles ought to have prevented the patentability of, or at least the enforceability of patents on, the Internet's basic protocols.[87]

In addition, patent law ought to open up examination and postexamination to third-party challenges, particularly as to the novelty and nonobviousness of software inventions. Glynn S. Lunney, Jr., suggests that there currently may be undetected obviousness in many a patented software invention, in that the relevant prior art often exists only in other software code or informal documents, making it nearly impossible for patent examiners to find and causing patents to issue erroneously.[88] For that reason, allowing third parties to suggest prior-art references relevant to an invention's novelty and nonobviousness would be helpful to ensure that software patents issue only for those inventions that are sufficiently innovative, thereby invigorating competition in the software industry. Moreover, patent examiners with relevant expertise in software would be particularly critical to evaluating the effect of this prior art on an invention's novelty and nonobviousness, as well as how straightforward it was to take

85. *See* Rochelle Cooper Dreyfuss, *Are Business Method Patents Bad for Business?*, 16 SANTA CLARA COMPUTER & HIGH TECH. L.J. 263, 271–72 (2000); Mark A. Lemley & David McGowan, *Legal Implications of Network Economic Effects*, 86 CAL. L. REV. 479, 538–40 (1998).

86. Cohen & Lemley, *supra* note 81, at 21–26.

87. *Cf.* John R. Allison & Emerson H. Tiller, *The Business Method Patent Myth*, 18 BERKELEY TECH. L.J. 987, 1009 n.60 (2003) (noting that, even if software had been patentable at the time, the protocols at the heart of the Internet likely would not have been patented, as many were developed by the government or collaboratively among many educational institutions).

88. Glynn S. Lunney, Jr., *E-Obviousness*, 7 MICH. TELECOMM. & TECH. L. REV. 363 (2001). That said, Allison and Mann demonstrate that, judged by objective factors, such as the number of prior art references and forward citations, software patents are indistinguishable—or perhaps stronger—than other patents and these factors hold whether the patentee is a large company or a small one. John R. Allison & Ronald J. Mann, *The Disputed Quality of Software Patents*, 85 WASH. U. L. REV. 297, 298 (2007).

what was oftentimes a real world business model and encode it in software.[89]

Ensuring the courts' discretion to mold patent law's general requirements to the specifics and context of any particular software patent would also help. As one example, Cohen and Lemley propose that because of software reuse and incremental innovation, infringement liability ought to be narrowly limited to those cases in which other software falls within the literal language of the patent's claims, not under patent law's broader doctrine of equivalents.[90] As another example, district courts have the discretion whether to enjoin infringers from using a patented invention or merely to require payments of damages.[91] If courts choose just to require damages, they are effectively awarding a compulsory license to the infringer: infringe again if you must, but you will have to pay. The more essential that incremental development is to a particular area of software, the more courts ought to feel compelled to deny injunctive relief, effectively implementing a scheme akin to compulsory licensing. Additionally, damages could decrease as patent duration wears on to reflect the combination of incremental software innovation and the fast pace of advancement in the field. Courts should have similar authority to tailor the novelty, nonobviousness, utility, and disclosure requirements to the particulars of software innovation as it unfolds and the industry evolves.[92]

None of these suggestions is exhaustive; rather, these suggestions are meant to indicate how the capacious patent law created by Congress can be tailored to the software industry's specifics, particularly in light of the Internet's key goals of generativity and end-to-end computing. In that sense, patent law is not that different from the Internet itself. Patent law itself is decentralized, in the sense that instantiations of patent law can often look very different as different firms build up varied patenting practices, whether through cross-licensing, low patenting rates, or patents as signals for funding. And patent law is highly generative, in the sense of being adaptable to the particulars of a myriad of technologies.[93] Patent law therefore surely has the capacity, if adapted properly, to promote innovation in Internet software.

89. Dreyfuss, *supra* note 85, at 278–79; Fromer, *supra* note 82, at 86–87, 97–98 (discussing how Amazon.com's patent on one-click shopping is likely obvious to software programmers, while computational-linguistics software frequently is not, even if humans already speak natural languages fluently).

90. Cohen & Lemley, *supra* note 81, at 45–53.

91. eBay Inc. v. MercExchange, L.L.C., 547 U.S. 388 (2006).

92. *Cf.* DAN L. BURK & MARK A. LEMLEY, THE PATENT CRISIS AND HOW THE COURTS CAN SOLVE IT (2009) (suggesting that courts ought to employ their discretion to tailor patent law to various technologies and industries, depending on their salient characteristics).

93. *Id.*